BROADCAST COMPRESSED FIRMWARE FLASHING

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is related to co-pending application Serial No. _____ (Att'y. Docket No. 1662-39300) titled "Intelligent Power Management for a Rack of Servers." This application is also related to co-pending application Serial No. _____ (Att'y. Docket No. 1662-39100) titled "Redundant Data and Power Infrastructure for Modular Server Components in a Rack."

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[0002] Not applicable.

BACKGROUND OF THE INVENTION

Field of the Invention

[0003] The present invention relates generally to flashing electrically erasable programmable read-only memories (EEPROMs). More particularly, the preferred embodiments of the present invention are directed to broadcasting images to be flashed to multiple EEPROMs, where the broadcasting occurs to each system supporting the EEPROM flashing substantially simultaneously. Also, the preferred embodiments are directed to reducing the size of the image to be flashed in the broadcast stage of the image transfer by predefining a frequency table for Huffman encoding, and using that frequency table thereafter for encoding and decoding each image transferred. In this way, the Huffman frequency table need not be transferred with the image.

Background of the Invention

[0004] Every computer system has memory devices with varying degrees of volatility. For example, almost every computer system has random access memory (RAM) for use by the microprocessor in temporarily storing data and programs. However, the contents of the RAM are lost when the computer system is powered down. Computers likewise have some kind of read-only memory (ROM). This ROM generally contains system boot information and basic input/output system (BIOS) programs. The contents of a standard ROM are not lost with a cycle of the power of the computer. This ROM could comprise many different types of read-only memory including ROM that may only be written one time (programmable read-only memory (PROM)), and various PROMs such as ultraviolet erasable PROMs (UVPROMs) and electrically erasable PROMs (EEPROMs). A particular kind of PROM known as flash EEPROM has found particular acceptance. The "flash" portion of the description ostensibly coming from the fact that this particular kind of EEPROM may be erased and rewritten relatively quickly, as compared to a UVPROM, which must be removed from the computing device and placed under ultraviolet light just to be erased.

[0005] Flashing EEPROM in the context of computer systems may also take place quickly because of a relatively high bandwidth communication buses between computer system components. Thus, when an EEPROM needs to be flashed, *e.g.*, to update the BIOS, the new software image is transferred from a storage medium of the computer, *e.g.*, a floppy drive, to the EEPROM. The time it takes to transfer the new software image over the high bandwidth bus is relatively insignificant. However, problems arise when the communication pathway between the long-term storage medium and the EEPROM is of low bandwidth. The problems arise in the amount of time it takes to transfer the image across that low bandwidth pathway.

[0006] Further, there may be multiple flash EEPROMs that need to be updated coupled to the low bandwidth bus. The amount of time to transfer the image to be flashed thus increases considerably given that the prior art method of transferring these images are to transfer them one at a time.

[0007] Thus, what is needed in the art is a way to minimize the amount of time it takes to transfer software images to remote EEPROMs across a low bandwidth bus, especially where the new image is used to flash each downstream EEPROM, and also where the new images are related to the old images on the EEPROMs.

BRIEF SUMMARY OF THE INVENTION

[0008] The problems noted above are solved in large part by a system and related method for sending images to be written to EEPROMs. In one aspect of the invention, where each of a plurality of EEPROMs is to receive the same image, the preferred embodiments comprise broadcasting the new image to each of the EEPROM substantially simultaneously. In this way, the time it takes to transfer the images to the EEPROMs is significantly reduced over the prior art technique of transferring each image individually.

[0009] In a second aspect of the invention, the image to be broadcast to each of the EEPROMs is compressed prior to being sent. In the preferred embodiments, this compression is done by Huffman encoding. However, the preferred embodiments of the present invention see gains over standard Huffman encoding by predefining a frequency table, and using that frequency table thereafter for each Huffman encoded compression. Since the frequency table is predefined, that frequency table need not be transferred with the Huffman encoded image. In this way, the overall compressed image to be transferred is smaller than standard Huffman encoding.

[0010] The preferred implementation of the two aspects of the preferred embodiment described above are in systems where EEPROMs that need new flash images lie downstream of a source of those images across a low bandwidth communication bus or pathway. Broadcasting the new image to the multiple EEPROMs, as well as compressing that image, saves substantial time in the transfer of the image prior to actually flashing in such a system. In a particular embodiment, the source of the new image is a server in a rack mounted computer system. In such a system, the EEPROMs to be flashed are EEPROMs containing software or firmware for communication modules in a communication system dedicated to control of central power supplies for the rack mounted system. The low bandwidth communication bus is an I^2C bus extending between the server and at least one communication module whose EEPROM needs to be updated.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] For a detailed description of the preferred embodiments of the invention, reference will now be made to the accompanying drawings in which:

[0012] Figure 1 shows a rack mounted computer system for a specific implementation of the broadcast compressed firmware flashing system;

[0013] Figure 2 shows a communication module of the specific implementation;

[0014] Figure 3 shows an exemplary frequency table for Huffman encoding;

[0015] Figure 4A shows a partial flow diagram of the method of broadcasting the images to the multiple EEPROMs; and

[0016] Figure 4B shows the remaining steps of the flow diagram for broadcasting the images.

NOTATION AND NOMENCLATURE

[0017] Certain terms are used throughout the following description and claims to refer to particular system components. As one skilled in the art will appreciate, computer companies may refer to a component by different names. This document does not intend to distinguish between components that differ in name but not function.

[0018] In the following discussion and in the claims, the terms "including" and "comprising" are used in an open-ended fashion, and thus should be interpreted to mean "including, but not limited to...". Also, the term "couple" or "couples" is intended to mean either an indirect or direct electrical connection. Thus, if a first device couples to a second device, that connection may be through a direct electrical connection, or through an indirect electrical connection via other devices and connections.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0019] It has been proposed in the computer industry to remove the AC-DC power supplies from the individual servers in the rack mounted server systems and centralize those power supplies. However, physically separating the power supplies from the computers that need that power requires a structure and method to allocate and de-allocate power. Co-pending application Serial No. _____, titled "Intelligent Power Management for a Rack of Servers" (Att'y. Docket No. 1662-39300), incorporated herein by reference as if reproduced in full below, describes a system to perform the allocation and de-allocation of power.

[0020] Figure 1 shows an implementation of a rack mounted server system 100 which is a specific implementation of the preferred embodiments, and which rack mounted system is of the kind described in the co-pending patent application titled "Intelligent Power Management for a Rack of

Servers." In particular, Figure 1 shows a plurality of chassis 20, each chassis having a plurality of servers 30 therein. Further, Figure 1 shows a power supply system 40 comprising two power supply assemblies 42. Each power supply assembly 42 preferably houses a plurality of individual power supplies 44.

[0021] Each chassis 20 preferably has associated therewith a chassis communication module 80. The chassis communication module 80 in each chassis 20 couples to each individual server 30 across a serial communication bus 82, which in the preferred embodiment is an I^2C bus. The I^2C bus 82 is a dual line, multidrop serial bus developed by Phillips Semiconductor that comprises a clock line and one data line. The devices connected to the I^2C bus can act as either primary or secondary devices, and each device is software addressable by a unique address. Primary devices can operate as transmitters, receivers, or combination transmitter/receivers to initiate eight-bit data transfers between devices on the bus. The I^2C bus utilizes collision detection and arbitration to prevent data corruption if two or more primaries simultaneously transfer data. Details regarding the I^2C bus may be found in "The I^2C-Bus Specification," version 2.1 (January 2000), authored by Phillips Semiconductors.®

[0022] Each chassis communication module preferably communicates with the servers 30 in its associated chassis over the I^2C bus 82. The chassis communication modules 80 also preferably couple to the power supply communication modules 70 in the power supply system 40. The chassis communication modules 80 and the power supply communication module 70 preferably couple by way of a serial communication pathway 60 being an Institute of Electrical and Electronic Engineers (IEEE) RS-485 bus.

[0023] Much like the chassis communication module 80, each power supply communication module preferably couples to a plurality of devices in its associated power supply assembly 42. In

particular, the power supply communication module 70 couples to each individual power supply 44 in its associated power supply assembly 42 over an I^2C serial bus 72. The power supply communication module may therefore monitor various parameters associated with the individual power supplies 44, including the power output of each individual power supply 44. By polling each individual power supply 44 across the I^2C bus 72, each power supply communication module 70 may make a determination as to remaining power capacity, if any, in its respective power supply assembly 42. Figure 1 shows two power supply assemblies 42, and two power supply communication modules 70. Preferably each of these power supply communication modules 70 has the ability to monitor parameters of the individual power supplies 44 within its respective power supply assembly 42. Preferably, however, only one of the power supply communication modules 70 is designated as the primary power supply communication module, which makes that primary power supply communication module responsible for communicating on behalf of the entire power supply system 40.

[0024] Thus, the chassis communication modules 80 transmit requests for allocation of power to the primary power supply communication module 70 over the serial communication pathway 60. Likewise, the primary power supply communication module 70 responds to those requests by sending messages across the serial communication pathway 60 to the requesting server 30, by way of its respective chassis communication module 80.

[0025] Figure 2 shows a chassis communication module 80 of the preferred embodiment. In particular, Figure 2 shows that the chassis communication module 80 comprises a digital computing means which in the preferred embodiment is a microcontroller 82, preferably a part No. ZIRCON-LH2 manufactured by QLOGIC Corporation; however, equivalent systems could be constructed using any available microcontroller or microprocessor. The chassis communication

module 80 also preferably comprises memory means being a random access memory array (RAM) 84 and an electrically erasable programmable read only memory (EEPROM) 86 coupled to the microcontroller. The RAM 84 is preferably working space for executing programs by the microcontroller. The EEPROM 86 preferably stores firmware programs that when executed by the microcontroller 82 perform the various functions required of the chassis communication module 80. Figure 2 also shows that the microcontroller 82 of the chassis communication module 80 preferably couples to the I^2C bus and the RS-485 bus. For purposes of this disclosure, each power supply communication module 70 has substantially the same components as a chassis communication module 80. Thus, Figure 2 showing the microcontroller 82, RAM 84 and EEPROM 86 for the chassis communication module 80 is equally applicable to the power supply communication module 70.

[0026] Of particular importance for this invention is the EEPROM 86 of Figure 2 (and the corresponding EEPROM in each of the power supply communication modules 70). This EEPROM preferably stores firmware programs that are executed by the microcontroller, which programs direct the microcontroller to perform the various functions required of the power supply communication module 70 and the chassis communication module 80. However, from time to time it is necessary to update that software, *e.g.*, to increase functionality or to fix bugs in the software. For simplicity and ease of manufacture, each of the chassis communication modules 80 and power supply communication modules 70 only communicate via either the I^2C bus or the RS-485 bus. That is, preferably there is no external port for accessing or communicating with the microcontroller other than these two serial communication pathways.

[0027] Thus, when the programs stored in the EEPROMs of the power supply communication module 70 and the chassis communication module 80 must be updated, that update preferably

takes place by means of an individual server 30 initiating a data transfer across the I^2C bus 82. In the case of the chassis communication module 80, the transfer preferably also takes place across the serial communication pathway 60. In the preferred embodiment, the I^2C bus 82 is clocked at 50 kilohertz, and assuming a one bit transfer with every clock cycle and no packet overhead in the data transfer, 50,000 bits may be transferred across the I^2C bus each second. Considering that each EEPROM image (the total file to be flashed onto the EEPROM) may be as large as two megabytes, the transfer of each image alone in the ideal case would take just over five minutes. As one of ordinary skill in the art is aware, data communication systems do not reach 100% packet efficiency; and therefore, a data transfer of two megabytes on a dedicated I^2C bus would most like take closer to twenty minutes (assuming a 25% data transfer efficiency). To complicate matters further, in the preferred embodiment the transfer of the new image from a server 30 to a chassis communication module 80 preferably takes place while the chassis communication module 70 is in normal operation. Thus, the complete bandwidth of the I^2C bus 82 will not be dedicated to the transfer, therefore increasing the time further. If we assume for purposes of explanation, and not as a limitation, that roughly half of the bandwidth of the I^2C bus 82 is dedicated to normal communication, it is clearly seen that the data transfer of a single image from a server 30 to a chassis communication module 80 may take forty minutes or more. Considering that in the preferred embodiment as many as six chassis 20 may be installed in a rack mounted server system 100, it could take as many as four hours to use traditional methods of transmitting software images -- one at a time.

[0028] Additionally, the preferred embodiment of the rack mounted server system also has two power supply communication modules 70. Thus, the software images for these devices preferably transfer from a server 30 across the I^2C bus 82, and then across the serial communication

pathway 60 to the power supply communication module 70. While the serial communication pathway 60 is preferably an RS-485 bus and thus has a greater bandwidth than the I^2C bus 82, the limiting factor is the transfer from the server 30 to the chassis communication module across the I^2C bus 82. Additional delays are encountered in the relaying of information by one of the chassis communication modules 80. Thus, it is seen that to transfer software images to each chassis communication module 80 and each power supply communication module 70 in a rack mounted computer system 100 using prior techniques itself could take as much as four and a half hours or more.

[0029] In broad terms, the preferred embodiments of this invention reduce the amount of time to complete the transfer of the software images and flashing of those images onto the various EEPROMs in the system by capitalizing on the fact that each of the power supply communication modules 70 preferably have the same software. Likewise, each of the chassis communication modules 80 preferably have the same software. In the preferred embodiment, a single image is broadcast, and each chassis communication module (for the broadcast of that software image) reads the new image. Likewise, each power supply communication module 70 reads the image during a broadcast of that image by a server 30. In this way, the prior art technique of sending the software image for each particular EEPROM is reduced to only a single broadcast transmission (with error recovery as discussed below). The efficiency of this system thus increases with each additional duplicate EEPROM to be flashed.

[0030] Before continuing, it must be understood that the preferred embodiments of this invention, while being developed and having a specific implementation in the context of the rack mounted server system having a central power supply system, are applicable to any system in which multiple flash EEPROMs lie along communication pathway. While the preferred embodiments

will be described in the context of such a rack mounted computer system, this should not be read as a limitation on the applicability of the structure and methods described herein.

[0031] Still speaking in broad terms, the preferred embodiments of this invention also reduce the size of the image broadcast (in either the chassis communication module case or the power supply communication case) by Huffman encoding. While one of ordinary skill in the art is aware of Huffman encoding, the preferred embodiment obtains gains over standard Huffman encoding by using a predetermined or predefined frequency table, and then just the Huffman encoded information is transferred (as opposed to both the encoded information and the frequency table). This technique is discussed more fully below. Thus, transferring images from a server 30 across a low bandwidth bus 82 is optimized by broadcasting the image to each device that needs that image, as well as compressing that image prior to transfer.

[0032] Huffman encoding is a technique for compressing information that relies on the fact that particular symbols of the uncompressed file have varying probabilities of occurrence within that file. Consider for purposes of explanation a source file having six unique symbols (A_1, A_2, ... A_5), with each symbol having varying degrees of probability of occurrence within the file. Further assume that the symbol A_1 has the highest probability of occurrence, symbol A_2 has the next highest probability of occurrence and so on through A_6 having the least probability of occurrence within the file. In Huffman encoding, because the symbol A_1 has the highest probability of occurrence within the file, it is assigned the smallest, in terms of number of bits, code within the coding scheme. Referring to Figure 3, there is shown a diagrammatic example of a Huffman frequency table used to decode and encode messages. Because it was assumed that the symbol A_1 had the highest probability of occurrence, within the table of Figure 3 the symbol A_1 is represented by a single bit, and arbitrarily assigned logic "1." The second-most probable symbol in the

exemplary file is A_2. In the exemplary Huffman encoding shown in Figure 3, the symbol A_2 is assigned a code logic "00." Thus, in Huffman encoding, a table such as that shown in Figure 3 is used to decode a series of zeros and ones which contain the encoded string. As an example, and without limitation, assume that the Huffman encoded string is as follows: 010100111100. Referring still to Figure 3, the decoding process starts at the start location 150. Because the first bit in the Huffman encoded string is zero, decoding this string proceeds along the zero path to point 152. Because the next bit in the Huffman encoded string is logic "1," the decoding proceeds along the logic "1" path from point 152 to point 154. Likewise, the next bit is a logic "0," which means the decoding process proceeds to point 156. The next bit is a logic "1" meaning that the decision process moves to point 158. Finally, the next bit in the Huffman encoded string is a logic "0," which means that the first symbol represented in the Huffman encoded system is A_5. Since we have reached the end of the table and determined the first symbol, the process starts again at start location 150. Continuing in the exemplary code just after the bit that identified the first symbol, the next bit encoded is a "0" so the decode process moves to point 152. The next bit is a "1," forcing the decode process to move to point 154. The next bit is a "1," which leads to the symbol A_3. Thus, the second bit in the Huffman encoded string is A_3. Starting again at block 150, the next bit encountered is a "1" which leads directly to the symbol A_1. Starting again with the next bit, again being a "1," this leads directly to the symbol A_1 again. Finally, starting in block 150 with the next bit "0," the decoding process moves to point 152. The final bit in the Huffman encoded string is a "0," thus leading to the symbol A_2. Thus, the 12 bits in our exemplary Huffman encoded string 010100111100 decodes to symbols A_5, A_3, A_1, A_1 and A_2 respectively. It can be clearly seen that if each one of these symbols A_1 ... A_5 are one byte of information (eight bits), 48 bits of information would have to be sent. Using the Huffman encoding, only 12 bits were required to

represent the same information. However, in standard Huffman encoding, the information required to decode the encoded message must also be transferred. In the exemplary table of Figure 3, 240 bits are required to represent the table. Thus, using standard Huffman, both the encoded message (in the exemplary system 12 bits) and the frequency table (in the exemplary system 240 bits) must be sent.

[0033] Thus, the preferred embodiment uses Huffman encoding to reduce the size of the images sent from the server 30 across the I^2C bus 82 to the chassis communication module 80. Likewise, the preferred embodiments use Huffman encoding to decrease the size of the image sent from a server 30 across an I^2C bus 82 and further across a serial communication pathway 60 to a power supply communication module 70. However, the preferred embodiments also achieve performance gains over pure Huffman encoding by using a predefined frequency table (an example of which is shown diagrammatically in Figure 3). Each of the receiving chassis communication modules 80 and power supply communication modules 70 preferably already have the frequency table, and thus the table need not be sent.

[0034] As has been discussed, the process of flashing the power supply communication module 70 and chassis communication module 80 flash EEPROM will generally take place to update the firmware held in the EEPROMs of each of those devices. It is envisioned that while these updates will increase the functionality and/or repair bugs in the code therein, the symbols required to make up that software preferably remain the same. In the preferred embodiments, a frequency table for the Huffman encoding is initially created based on the original programs stored on the EEPROM. That frequency table for Huffman encoding is preferably hard-coded into the EEPROM, and thereafter is used for each update or flash of the EEPROM. Stated otherwise, the initial frequency table for Huffman encoding will most likely not change significantly with minor updates of the

software thereon. The preferred embodiments of the present invention shrink the size of the file that must be transferred across the low bandwidth bus by using the same frequency table for each image transfer. Thus, this frequency table need not be transferred; instead, preferably only the Huffman encoded information is sent across the low bandwidth bus. It is acknowledged that in some cases the probability of occurrence of a particular symbol may change in an image to be sent; however, those changes of probability for particular symbols will most likely be limited. Thus, while it may be possible in subsequent Huffman encoded transfers to shrink the size of the image further if a new frequency table were used, those gains would be lost in having to transfer the frequency table every time.

[0035] Thus, the preferred embodiments hard-code a predefined Huffman frequency table, and thereafter use that frequency table in decoding images sent across the low bandwidth bus. Likewise, the source of the image, preferably a server 30 within the rack mounted system 100, preferably Huffman encodes each image using that initial frequency table, even if minor changes may have occurred with respect to the probabilities of occurrence of any particular symbol.

[0036] Referring now to Figures 4A and 4B, there is shown a flow diagram of the preferred method for updating the flash EEPROM in multiple devices across a low bandwidth bus. In particular, the process starts at block 200 and the first step is the activation of each receiving microcontroller, as indicated at block 210. In the preferred embodiments, at least part of the image to be flashed is transferred during normal operation of the communication module (be it the chassis communication module 80 or the power supply communication modules 70). Thus, this step informs each receiving microcontroller of the upcoming broadcast of the new image so that each microcontroller to be updated will be aware of its responsibility to receive and store the packets of information. Once each receiving microcontroller has been activated, preferably the initiating

server 30 sends portions of the flash image in packet form as a broadcast message across the network, as indicated at block 220. The process of sending the image in broadcast form continues in small portions until the entire image has been broadcast across the network, as indicated by the combination of blocks 220 and 230.

[0037] Once the entire image has been broadcast by the initiating server 30, that server preferably then sends a request to each microcontroller in each communication module to send an identification of parts of the image that each particular microcontroller did not receive, as indicated in block 240. Because the broadcast of the image preferably takes place during normal operation of the communication modules, it is possible that some of the information may be lost due to collisions of data on one or both of the I^2C bus 82 or the serial communication pathway 60. Also, it is possible that a particular microcontroller of a particular communication module may have missed recording a portion of the image broadcast because it was busy servicing other requests. Regardless of the reason, before flashing the image into the ROM, it is necessary to have the complete image. Thus, the step as indicated at block 240 informs the initiating server 30 of those packets missing from each communication module. Thus, if any parts are missing as indicated in block 250, those missing components of the image are sent again in broadcast form as indicated block 260.

[0038] This process of sending portions of the image that were not received preferably continues until each microcontroller contains the complete image to be flashed in the RAM associated with the microcontroller. With regard to sending those missing pieces of the image, it will be understood that if more than one microcontroller is missing a single piece of information, the initiating server preferably transmits that missing piece of information only once in broadcast form (similar to the initial image broadcast). In other words, the gaps in each of the images held by each

microcontroller are filled in broadcast form, and the efficiency of this process is increased if multiple microcontrollers are missing the same piece of information.

[0039] Once each microcontroller has a complete copy of the image to be flashed, the initiating server preferably sends a request to a first microcontroller to decode and flash the image as indicated in block 270 of Figure 4B. This request, however, is not a broadcast request; but instead, is a specific request for the first microcontroller to flash the image. If the communication module successfully boots after the image flash as indicated in block 280, the process continues to each subsequent microcontroller in each subsequent communication module, as indicated at block 290. If a particular communication module successfully flashes and boots, and it was the last communication module, as indicated at decision block 290, then the process notifies the user that the updates are complete as indicated in block 300, and the process ends. If, however, any communication module fails to properly boot after the firmware flash, the error is reported and subsequent flashing halted, as indicated in block 310. This halting of further flashing is to prevent all modules from becoming inoperable if the image is not stable.

[0040] As discussed above, in the preferred embodiments, the broadcast of the image to be flashed is preferably accomplished during normal operations of each of the communication modules. This preferred implementation assumes that each module has sufficient RAM to store the image (which may be up to 2 megabytes), and still have sufficient working memory for performing communication module duties. In the case where the image size is too large to allow sufficient working space, it is preferred that as much of the image as possible is transferred in the broadcast mode. Thereafter, each individual microcontroller in a particular communication module is preferably placed in a pre-load mode wherein normal operations are suspended and the

microcontroller is dedicated to receiving the remainder of the image, decoding and flashing that image.

[0041] The above discussion is meant to be illustrative of the principles and various embodiments of the present invention. Numerous variations and modifications will become apparent to those skilled in the art once the above disclosure is fully appreciated. For example, the preferred embodiments have been described with respect to a rack mounted computer system having a centralized power supply system apart from the individual servers in the rack. Thus, the method of broadcasting the images and flashing the multiple flash EEPROMs in the communication module across the relatively low bandwidth I^2C bus 82 was described with respect to this system. However, one of ordinary skill in the art, now understanding how the flashing takes place could easily design an equivalent system for any application where the flash EEPROMs are located down a communication bus with limited bandwidth such that transferring each individual image becomes prohibitive. Further, the flash EEPROMs of interest in this disclosure are those EEPROMs and communication modules controlled by microcontrollers; however, the system and methods described herein could equivalently be applied to any system in which software needs to be updated in remote devices. It is intended that the following claims be interpreted to embrace all such variations and modifications.